Math Clinic Status Update 1

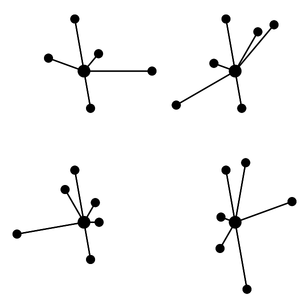
Team 1: Komi Agbo, Dalton Burke, Nick Mako, James Vance

October 28th, 2019

1 Since our last update...

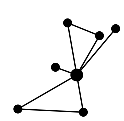
We have completely changed our approach to the problem. Instead of thinking of the problem as a Vehicle Routing Problem, where the order of deliveries directly effects the cost of the route, our team noticed that how much the landfills and storage locations are visited nearly eliminates this ordering constraint.

If customers only ever wanted container switches, the problem would be much simpler. Where should the truck come from that's going to deliver the new container? The nearest landfill. And where should the truck go after picking up the full container? The nearest landfill. Filling the route with this data, we end up with several *stars*, each landfill being a center, and the points surrounding it being the job sites. To service each star, you can take the jobs in any order, the resulting cost will be the same. This will give us a lot of leniency when trying to satisfy the constraints. The only task remaining is to sew them together into routes for the drivers.



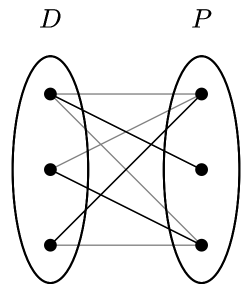
Stars

Customers can request drop-offs as well as pick-ups, with this, there are some opportunities for extra optimization. A driver can drop off a container, then, with the empty truck, go complete a pickup, without having to visit the landfill. A driver completing a route like this (landfill → drop-off → pick-up → landfill) makes a *triangle*, hence our name for this style, triangle method.



Triangles

A big question comes to mind, how do we choose the best triangles? Certainly, some are better than others. To make this decision, we can construct a bipartite graph, one set containing the drop-offs, and the other containing pick-ups, and the existence of an edge between a drop-off and pick-up means that they are *compatible* (their constraints can work together). The weight of the edge between a drop-off and a pick-up is calculated as the difference between doing each normally (star method) and doing them combined as a triangle. In this graph, a matching pairs up drop-offs and pick-ups to be completed as a triangle. A minimum matching in this graph gives us the best possible choice of triangles in order to make the route take the least amount of time. Minimum matchings are extremely easy to compute (at least, compared to traveling salesman type problems), our plan is to use the Hungarian Algorithm.



An example matching

2 What's on our plate now...

- Writing some basic code for the star case
- Working out some more details for
  - Choosing transitions between landfills
  - Satisfying constraints
  - Using Delivery-Pickup strings as transitions (when they are more optimal than normal transitions)

3 What's next...

- Write more integration with data from UI team
- Write D-P matching
- Work more with the details from the last section